Signal Processing: *Image Communication* 15 (2000) 559–565

SIGNAL PROCESSING: IMAGE COMMUNICATION

www.elsevier.nl/locate/image

On image compression by competitive neural networks and optimal linear predictors

Robert Cierniak*, Leszek Rutkowski

Department of Computer Engineering, Technical University of Czestochowa, 42-200 Czestochowa, ul. Armii Krajowej 36, Poland

Abstract

In this paper a new algorithm for image compression, named predictive vector quantization (PVQ), is developed based on competitive neural networks and optimal linear predictors. The semi-closed-loop PVQ methodology is studied. The experimental results are presented and the performance of the algorithm is discussed.

Keywords: Image compression; Vector quantization; Optimal linear predictor; Neural networks; Frequency-sensitive competitive learning

1. Introduction

In the last decade the vector quantization (VQ) technique [4,5] has emerged as an effective tool for image compression [6,12]. A special approach to image compression combines the VQ technique with traditional (scalar) differential pulse code modulation (DPCM) leading to the predictive vector quantization (PVQ). The design of the PVQ scheme requires both a predictor and a VQ codebook determination. Gersho and Grey [4, pp. 493–496] outlined three approaches to the design problem:

(i) open-loop design methodology (applied to image compression in [2])
(ii) closed-loop design methodology (applied to image compression in [2])
(iii) semi-closed-loop design methodology

Semi-closed-loop approach has never been elaborated in the context of image compression. In this paper it will be shown that the semi-closed-loop approach gives a useful improvement over the open-loop methodology. Obviously our methodology requires less computations than the closed-loop approach.

We present details of the semi-closed-loop methodology. The vector quantizer will be based on the competitive neural networks, whereas the predictor will be designed in an optimal way – contrary to a heuristic method presented in [3].

In the sequel we assume that an image is represented by an $N_1 \times N_2$ array of pixels y_{ij}, $i = 1, 2, \ldots, N_1$, $j = 1, 2, \ldots, N_2$.

The image is portioned into contiguous small blocks of the dimension $n_1 \times n_2$.

$$\boldsymbol{Y}(m,n) = \begin{bmatrix} y_{1,1}(m,n) & \cdots & y_{1,n_2}(m,n) \\ \vdots & \ddots & \vdots \\ y_{n_1,1}(m,n) & \ldots & y_{n1,n_2}(m,n) \end{bmatrix}, \qquad (1)$$

* Corresponding author. Tel.: 48-34-3250546; fax: 48-34-3256963.

E-mail address: cierniak@matinf.pcz.czest.pl (R. Cierniak)

0923-5965/00/$ - see front matter
PII: S0923-5965(99)00013-2

VISIT…

LANZAROTE Caliente.com

where $m = 1, 2, \ldots, N_1/n_1$ and $n = 1, 2, \ldots, N_2/n_2$. Obviously it is assumed that

$$\left(\frac{N_1}{n_1}\right) \bmod 1 = 0 \quad \text{and} \quad \left(\frac{N_2}{n_2}\right) \bmod 1 = 0. \tag{2}$$

The arrays (1) will be represented by the corresponding vectors

$$\boldsymbol{V}(m,n) = [v_1(m,n), v_2(m,n), \ldots, v_q(m,n)]^{\mathrm{T}}, \tag{3}$$

where we identify

$$v_1(m,n) = y_{1,1}(m,n),\ v_2(m,n) = y_{1,2}(m,n), \ldots, v_q(m,n) = y_{n_1,n_2}(m,n)$$

and

$$q = n_1 n_2.$$

That means that the original image is represented by $N_1 N_2/q$ vectors $\boldsymbol{V}(m,n)$.

2. The general architecture of the PVQ algorithm

The general architecture of the predictive vector quantization algorithm (PVQ) is depicted in Fig. 1. This architecture is a straightforward vector extension of the traditional (scalar) differential pulse code modulation (DPCM) scheme [7].

The block diagram of the PVQ algorithm consists of an encoder and decoder, each containing an identical predictor, codebook and vector quantizer. The successive input vectors $\boldsymbol{V}(m,n)$ are introduced to the encoder and the difference $\boldsymbol{E}(m,n) = [e_1(m,n), e_2(m,n), \ldots, e_q(m,n)]^{\mathrm{T}}$ given by the equation

$$\boldsymbol{E}(m,n) = \boldsymbol{V}(m,n) - \bar{\boldsymbol{V}}(m,n) \tag{4}$$

is formed, where $\bar{\boldsymbol{V}}(m,n) = [\bar{v}_1(m,n), \bar{v}_2(m,n), \ldots, \bar{v}_q(m,n)]^{\mathrm{T}}$ is the predictor of $\boldsymbol{V}(m,n)$. As in the scalar

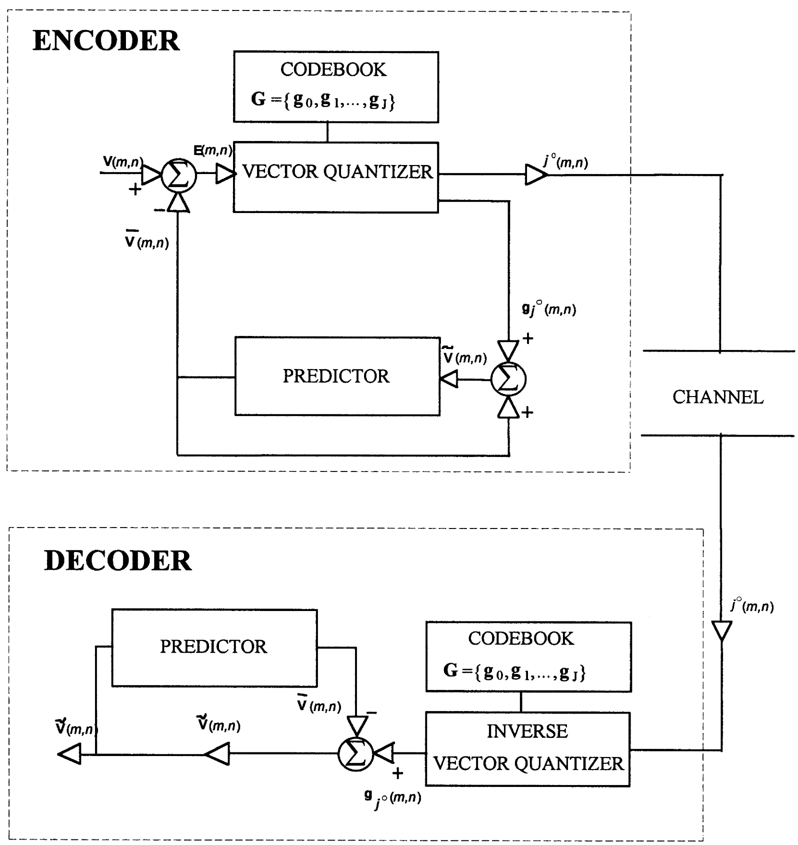

Fig. 1. Block diagram of the VQ DPCM image compression system.

DPCM, the difference $\boldsymbol{E}(m,n)$ requires fewer quantization bits than the original subimage $\boldsymbol{V}(m,n)$. The next step is vector quantization of $\boldsymbol{E}(m,n)$. Mathematically, the vector quantization can be viewed as a mapping $\boldsymbol{VQ}$ from the q-dimensional Euclidean space $\boldsymbol{R}^q$ into a finite subset $\boldsymbol{G}$ of $\boldsymbol{R}^q$,

$$\boldsymbol{VQ}\colon \boldsymbol{R}^q \to \boldsymbol{G}, \tag{5}$$

where

$$\boldsymbol{G} = [\boldsymbol{g}_0, \boldsymbol{g}_1, \ldots, \boldsymbol{g}_J] \tag{6}$$

is the set of reproduction vectors (codewords or codevectors)

$$\boldsymbol{g}_j = [g_{1,j}, g_{2,j}, \ldots, g_{q,j}]^{\mathrm{T}}. \tag{7}$$

The subset $\boldsymbol{G} \subset \boldsymbol{R}^q$ is commonly called the codebook. For every q-dimensional difference vector $\boldsymbol{E}(m,n)$, the distortion (usually the mean square error) between $\boldsymbol{E}(m,n)$ and every codeword $\boldsymbol{g}_j$, $j = 0, 1, \ldots, J$, is computed. The codeword $\boldsymbol{g}_{j^o}(m,n)$ is selected as the representation vector for $\boldsymbol{E}(m,n)$ if

$$d_{j^o} = \min_{0 \leqslant j \leqslant J} d_j, \tag{8}$$

where

$$d_j = \sqrt{\textstyle\sum_{i=1}^{q} [e_i(m,n) - g_{ij}]^2}. \tag{9}$$

The index $j^o(m,n)$ is broadcast via the transmission channel to the decoder. Mathematically, encoding is the mapping

$$\boldsymbol{R}^q \to j^o. \tag{10}$$

Observe that by adding the prediction vector $\bar{\boldsymbol{V}}(m,n)$ to the quantized difference vector $\boldsymbol{g}_{j^o}(m,n)$ we get the reconstructed approximation $\tilde{\boldsymbol{V}}(m,n)$ of the original input vector $\boldsymbol{V}(m,n)$, i.e.

$$\tilde{\boldsymbol{V}}(m,n) = \bar{\boldsymbol{V}}(m,n) + \boldsymbol{g}_{j^o}(m,n). \tag{11}$$

As a measure of error between the original and reconstructed images one can take the mean square error

$$\mathrm{MSE} = \frac{1}{N_1 N_2} \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} (y_{ij} - \tilde{y}_{ij})^2 \tag{12}$$

or a signal-to-noise ratio (in decibel units)

$$\mathrm{SNR} = 10 \log_{10}\left(\frac{(\max\{y_{ij}\})^2}{\mathrm{MSE}}\right), \tag{13}$$

where $\tilde{y}_{ij}$, $i = 1, 2, \ldots, N_1$, $j = 1, 2, \ldots, N_2$, stand for pixels of the reconstructed image. The prediction vector $\bar{\boldsymbol{V}}(m,n)$ of the input vector $\boldsymbol{V}(m,n)$ is made from past observations of reconstructed vectors $\tilde{\boldsymbol{V}}(m-k, n-l)$, $k = 1, 2, \ldots, K$ and $l = 1, 2, \ldots, L$. The predictor has the form

$$\bar{\boldsymbol{V}}(m,n) = \sum_{k=1}^{K} \sum_{l=1}^{L} \boldsymbol{A}_{kl} \tilde{\boldsymbol{V}}(m-k, n-l), \tag{14}$$

where each $\boldsymbol{A}_{kl}$ is a $q \times q$ matrix, K and L are horizontal and vertical prediction orders, respectively. In the decoder, the index $j^o(m,n)$ transmitted by the channel is inverse vector-quantized

$$\boldsymbol{VQ}^{-1}\colon j^o \to \boldsymbol{R}^q \tag{15}$$

and the reconstructed vector $\tilde{\boldsymbol{V}}(m,n)$ is formed in the same manner as in the encoder (see Eq. (11)).

3. The semi-closed-loop predictive vector quantization procedure

The design of a predictive vector quantization scheme requires both a predictor (14) and a codebook (6) design. In this paper we use the semi-closed-loop methodology as shown in Fig. 2, mentioned by Gersho and Gray [4]. This approach contains of three steps:

(a) design of the predictor based on the statistics of $\boldsymbol{V}(m,n)$,
(b) generation of an initial codebook,
(c) design of the codebook based on the fixed prediction residuals

$$\boldsymbol{E}(m,n) = \boldsymbol{V}(m,n) - \sum_{k=1}^{K} \sum_{l=1}^{L} \boldsymbol{A}_{kl} \tilde{\boldsymbol{V}}(m-k, n-l), \tag{16}$$

where $(k,l) \neq (0,0)$.

3.1. Design of the predictor – the covariance method

Since the horizontal–vertical direction vector linear prediction seems to be rather complicated we

start with a simpler unidirectional problem. Let $\{\boldsymbol{X}(t)\}$be a stationary random sequence of q-dimensional vectors with finite second moment, i.e. $E(\|\boldsymbol{X}(t)\|^2) < \infty$. The k-order predictor $\bar{\boldsymbol{X}}(t)$ of the current vector $\boldsymbol{X}(t)$ is given by

$$\bar{\boldsymbol{X}}(t) = \sum_{k=1}^{K} \boldsymbol{A}_k \boldsymbol{X}(t-k), \tag{17}$$

where each $\boldsymbol{A}_k$ is a $q \times q$ prediction matrix. The prediction residual vector is defined as

$$\boldsymbol{e}(t) = \boldsymbol{X}(t) - \bar{\boldsymbol{X}}(t). \tag{18}$$

A vector predictor is said to be optimal if it minimises the mean square error

$$D(t) = E\|\boldsymbol{X}(t) - \bar{\boldsymbol{X}}(t)\|^2. \tag{19}$$

The optimal coefficient matrices minimising the error measure (19) satisfy the normal equation

$$\begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} & \cdots & \boldsymbol{R}_{1K} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} & \cdots & \boldsymbol{R}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{R}_{K1} & \boldsymbol{R}_{K2} & \cdots & \boldsymbol{R}_{KK} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_1^{\mathrm{T}} \\ \boldsymbol{A}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{A}_K^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{10} \\ \boldsymbol{R}_{20} \\ \vdots \\ \boldsymbol{R}_{K0} \end{bmatrix} \tag{20}$$

or in the compact form

$$\boldsymbol{R}\boldsymbol{A} = \boldsymbol{r}, \tag{21}$$

where

$$\boldsymbol{R}_{kl} = E[\boldsymbol{X}(t-k)\boldsymbol{X}^{\mathrm{T}}(t-l)] \tag{22}$$

are the correlation matrices, $k = 1, 2, \ldots, K$; $l = 0, 1, \ldots, K$.

The matrix $\boldsymbol{R}$ is a square $qK \times qK$ "supermatrix". The supermatrix $\boldsymbol{R}$ is a block-Toeplitz matrix.

In the literature equation (20) is called the multichannel normal equation [8,11] and can be solved recursively by making use of a generalised version of the Levinson–Durbin algorithm.

In practice, correlation matrices (22) are not known, but we have empirical data $\boldsymbol{X}(1), \boldsymbol{X}(2), \ldots, \boldsymbol{X}(N)$. The matrices $\boldsymbol{A}_k$ are estimated by minimising an estimate of the mean square error (19),

$$\hat{D}(t) = \sum_{t \in T} \|\boldsymbol{X}(t) - \bar{\boldsymbol{X}}(t)\|^2 = \sum_{t \in T} \left\| \boldsymbol{X}(t) - \sum_{k=1}^{K} \boldsymbol{A}_k \boldsymbol{X}(t-k) \right\|^2. \tag{23}$$

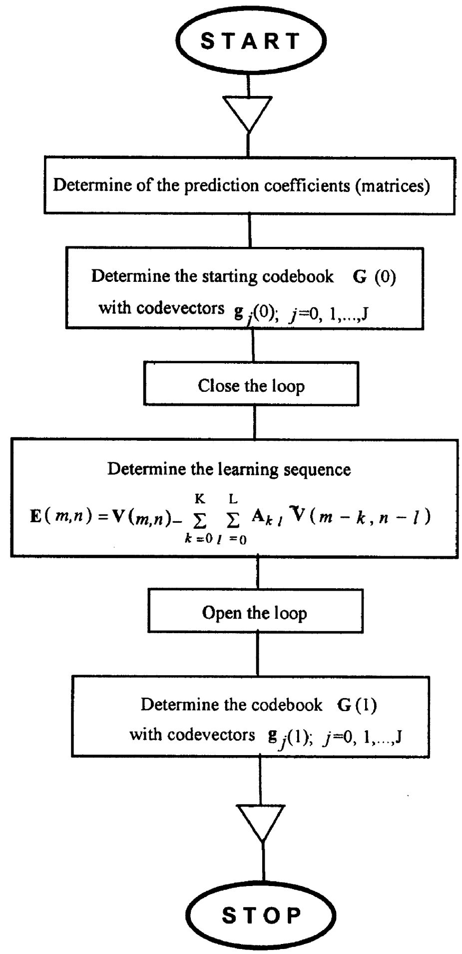

Fig. 2. Semi-closed-loop VQ design.

The estimates $\hat{\boldsymbol{A}}_K$ of matrices $\boldsymbol{A}_K$ satisfy the equation

$$\begin{bmatrix} \hat{\boldsymbol{R}}_{11} & \hat{\boldsymbol{R}}_{12} & \cdots & \hat{\boldsymbol{R}}_{1K} \\ \hat{\boldsymbol{R}}_{21} & \hat{\boldsymbol{R}}_{22} & \cdots & \hat{\boldsymbol{R}}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\boldsymbol{R}}_{K1} & \hat{\boldsymbol{R}}_{K2} & \cdots & \hat{\boldsymbol{R}}_{KK} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{A}}_1^{\mathrm{T}} \\ \hat{\boldsymbol{A}}_2^{\mathrm{T}} \\ \vdots \\ \hat{\boldsymbol{A}}_K^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{R}}_{10} \\ \hat{\boldsymbol{R}}_{20} \\ \vdots \\ \hat{\boldsymbol{R}}_{K0} \end{bmatrix}, \tag{24}$$

where

$$\hat{\boldsymbol{R}}_{kl} = \sum_{t \in T} [\boldsymbol{X}(t-k)\boldsymbol{X}^{T}(t-l)]. \tag{25}$$

The above methods can be extended to horizontal–vertical direction vector linear prediction. The

normal equation takes the form

$$\sum_{k=0}^{K}\sum_{l=0}^{L} \boldsymbol{A}_{kl}\boldsymbol{R}_{kl,ij} = \boldsymbol{R}_{00,ij} \tag{26}$$

for $i = 0, 1, \ldots, K$, $j = 0, 1, \ldots, L$, $(l, k) \neq (0, 0)$, $(i, j) \neq (0, 0)$,

where

$$\boldsymbol{R}_{kl,ij} = E[\boldsymbol{X}(m-k, n-l)\boldsymbol{X}^T(m-i, n-j)], \tag{27}$$

$$\boldsymbol{R}_{00,ij} = E[\boldsymbol{X}(m, n)\boldsymbol{X}^T(m-i, n-j)]. \tag{28}$$

If $K = 1$ and $L = 1$ then one gets

$$\begin{aligned} &\boldsymbol{A}_{10}\boldsymbol{R}_{10,10} + \boldsymbol{A}_{01}\boldsymbol{R}_{01,10} + \boldsymbol{A}_{11}\boldsymbol{R}_{11,10} = \boldsymbol{R}_{00,10},\\ &\boldsymbol{A}_{10}\boldsymbol{R}_{10,01} + \boldsymbol{A}_{01}\boldsymbol{R}_{01,01} + \boldsymbol{A}_{11}\boldsymbol{R}_{11,01} = \boldsymbol{R}_{00,01},\\ &\boldsymbol{A}_{10}\boldsymbol{R}_{10,11} + \boldsymbol{A}_{01}\boldsymbol{R}_{01,11} + \boldsymbol{A}_{11}\boldsymbol{R}_{11,11} = \boldsymbol{R}_{00,11}. \end{aligned} \tag{29}$$

In this case the autocorrelation and covariance methods are also applicable with an obvious modification.

3.2. Generation of an initial codebook

We will find the codebook $\boldsymbol{G}(0) = [\boldsymbol{g}_0(0), \boldsymbol{g}_1(0), \ldots, \boldsymbol{g}_J(0)]$ minimising the performance measure

$$D = \sum_{m=1}^{N_1/n_1}\sum_{n=1}^{N_2/n_2} d^2[\boldsymbol{E}(m, n), \boldsymbol{g}_{j^o}(0)], \tag{30}$$

where

$$d[\boldsymbol{E}(m, n), \boldsymbol{g}_{j^o}(0)] = \min_{0 \leqslant j \leqslant J} \{d[\boldsymbol{E}(m, n), \boldsymbol{g}_j(0)]\} \tag{31}$$

and d is the distortion (usually chosen as the mean square error) between the vector $\boldsymbol{E}(m, n)$ and the code vector $\boldsymbol{g}_j(0)$. The codevector $g_{j^o}(0)$ with the minimum distortion is called the "winner" [9]. The elements of the input vector $E(m, n) = [e_1(m, n), e_2(m, n), \ldots, e_q(m, n)]^{\mathrm{T}}$ are connected to every neural unit having the weight $\boldsymbol{W}_j = [w_{1,j}, w_{2,j}, \ldots, w_{q,j}]^{\mathrm{T}}$ and the output z_j, $j = 0, 1, \ldots, J$. The weights $\boldsymbol{W}_j$ are considered to be the codevectors, i.e.

$$\begin{aligned} \boldsymbol{G}(0) &= [\boldsymbol{g}_0(0), \boldsymbol{g}_1(0), \ldots, \boldsymbol{g}_J(0)]\\ &= [\boldsymbol{W}_0, \boldsymbol{W}_1, \ldots, \boldsymbol{W}_J] \end{aligned} \tag{32}$$

and the number of neural units $J + 1$ is the size of the codebook.

The weights $\boldsymbol{W}_j$ will be determined by making use of the neural networks. We apply two competitive learning neural networks:

(i) the competitive learning (CL) networks;
(ii) the frequency-sensitive competitive learning (FSCL) networks.

In case of the CL network [10,13] the distortion measure takes the form

$$\begin{aligned} d[\boldsymbol{E}(m, n), \boldsymbol{W}_j] &= \|\boldsymbol{E}(m, n) - \boldsymbol{W}_j\|\\ &= \sqrt{\textstyle\sum_{i=1}^{q}[e_i(m, n) - w_{ij}]^2}. \end{aligned} \tag{33}$$

The index j^o is selected such that

$$d[\boldsymbol{E}(m, n), \boldsymbol{W}_{j^o}] = \min_{0 \leqslant j \leqslant J} \{d[\boldsymbol{E}(m, n), \boldsymbol{W}_j]\}. \tag{34}$$

The output z_j of each unit is computed as follows:

$$z_j = \begin{cases} 1 & \text{for } j = j^o,\\ 0 & \text{for } j \neq j^o. \end{cases} \tag{35}$$

The new weight vector is computed as

$$\boldsymbol{W}_j(t+1) = \boldsymbol{W}_j(t) + \alpha[\boldsymbol{E}(m,n) - \boldsymbol{W}_j(t)]z_j, \tag{36}$$

where α is a learning parameter decreasing to zero as learning progresses. In the FSCL network [1,2] the winning neural unit j^o is selected on the basis of a modified distortion measure,

$$\begin{aligned} d[\boldsymbol{E}(m, n), \boldsymbol{W}_j] &= F(f_j)\|\boldsymbol{E}(m, n) - \boldsymbol{W}_j\|\\ &= F(f_j)\sqrt{\textstyle\sum_{i=1}^{q}[e_i(m, n) - w_{ij}]^2}, \end{aligned} \tag{37}$$

where F is a suitably chosen function of the counter f_j. The counter f_j counts how frequently the neural unit j is the "winner". The recursive procedure takes the form

$$\boldsymbol{W}_j(t+1) = \boldsymbol{W}_j(t) + H(f_j)[\boldsymbol{E}(m, n) - \boldsymbol{W}_j(t)]z_j, \tag{38}$$

where H is another function of the counter f_j.

3.3. Generation of the codebook

After determination of the codebook $\boldsymbol{G}(0) = [\boldsymbol{g}_0(0), \boldsymbol{g}_1(0), \dots, \boldsymbol{g}_J(0)]$ using the open-loop design we will compute the set of the $\boldsymbol{E}(m, n)$ vectors using the closed-loop scheme. This set is necessary to design the codebook $\boldsymbol{G}(1) = [\boldsymbol{g}_0(1), \boldsymbol{g}_1(1), \dots, \boldsymbol{g}_J(1)]$. Again we apply the competitive neural network described in Section 3.2 in order to find the codebook $\boldsymbol{G}(1)$.

4. Experimental results

The tested image was a standard picture "Lena" ($N_1 \times N_2 = 512 \times 512$ frame of size, 256 grey levels

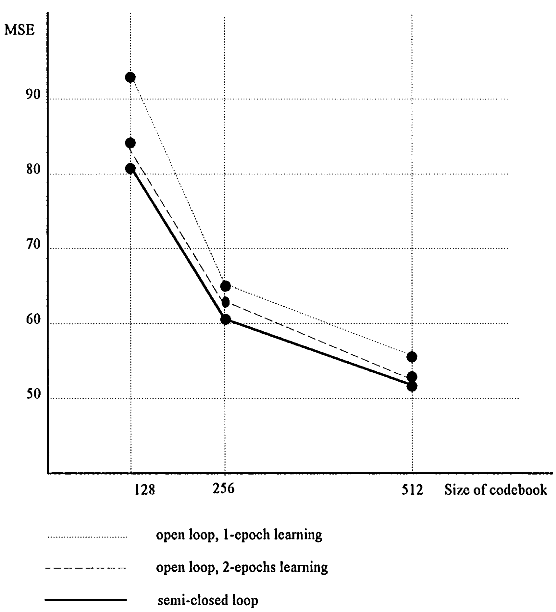

Fig. 3. The comparison of three compression methods.

Table 1
The comparison of the three compression methods using the MSE

Codebook size	128	256	512
Semi-closed loop	80.66	60.13	52.97
Open loop (1 epoch)	92.40	64.50	55.29
Open loop (2 epochs)	83.63	62.82	54.02

Fig. 4. (a) "Lena" original image; (b) "Lena" reconstructed image; (c) Difference between the reconstructed and original images.

for each pixel and blocks of image 4×4 pixels) as shown in Fig. 4(a). The experiment compares two methods of the realisation of the PVQ design: open-loop [8] and semi-closed-loop algorithms. Fig. 3 shows the results of the experiment for varying codebook size. The semi-closed-loop algorithm yields better results than the open-loop algorithm. Because for the semi-closed-loop algorithm the start codebook was designed using open-loop algorithm, the open-loop algorithm with learning for two epochs is more comparable. The results of these simulations are summarised in Table 1.

The experiment indicates that the FSCL (the frequency-sensitive competitive learning [1,2]) algorithm applied to the initial codebook determination gives the best results. The following functions were selected in the FSCL algorithm:

$$F(f_j) = 1 - \mathrm{e}^{-f_j/700}, \quad H(f_{j^o}) = 0.1 \cdot \mathrm{e}^{-f_{j^o}/1000}.$$

The counter f_j counts how frequently the neural unit j is the "winner", F is a suitably chosen function of the counter f_j, H is the learning coefficient.

To determine the final codebook the learning algorithm CL (the competitive learning) was used with the decreasing learning coefficient.

Fig. 4(b) shows the performance of the compression algorithm (blocks of image 4×4 pixels, the size of the codebook: 128).

5. Final remarks

We have presented the semi-closed-loop design methodology in the predictive vector quantization problem. Our methodology overperforms the open-loop methodology and requires less computations than the closed-loop approach. The codebook was designed based on the competitive learning neural networks including the FSCL method giving the best result. Contrary to the previous similar approach [3] our predictor is chosen in an optimal way, what results in better compression quality.

References

[1] S.C. Ahalt, A.K. Krishnamurthy, P. Chen, D.E. Melton, Competitive learning algorithms for vector quantization, Neural Networks 3 (1990) 227–290.

[2] R. Cierniak, Application of neural networks to image compression, Ph.D. dissertation, 1995.

[3] J.E. Fowler, M.R. Carbonara, S.C. Ahalt, Image coding using differential vector quantization, IEEE Trans. Circuits Systems Video Technol. 3 (1) (1993) 350–367.

[4] A. Gersho, R.M. Gray, Vector Quantization and Signal Compression, Kluwer Academic Publishers, Boston, 1992.

[5] R. Gray, Vector quantization, IEEE ASSP Mag. (1984) 4–29.

[6] H.M. Hang, J.W. Woods, Predictive vector quantization of images, IEEE Trans. Commun. 33 (1985) 1208–1219.

[7] A.K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[8] S.A. Kay, Modern Spectral Estimation, Prentince-Hall, Englewood Cliffs, NJ, 1988.

[9] T. Kohonen, Self-Organization and Associative Memory, Springer, Berlin, 1988.

[10] C.C. Lu, Y.H. Shin, Neural networks for classified vector quantization of images, Eng. Appl. Artif. Intell. 5 (5) (1992) 451–456.

[11] S.L. Marple, Digital Spectral Analysis with Application, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[12] N.M. Nasrabadi, R.A. King, Image coding using vector quantization: a review, IEEE Trans. Commun. 36 (8) (1988) 957–971.

[13] L. Rutkowski, R. Cierniak, Image compression by competitive learning neural networks and predictive vector quantization, Appl. Math. Comput. Sci. 6 (3) (1996) 431–445.